National College of Ireland

BSc in Computing

2015/2016

Niall O'Connell

X12431302

X12431302@student.ncirl.ie

Campus Talk

Technical Report

National
College *of*
Ireland

Name: **<u>Niall O’Connell</u>**
Student ID: **<u>x12431302</u>**
Supervisor: **<u>Paul Hayes</u>**

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: ____

Date: ___________

Table of Contents

Executive Summary

The aim of this application is to provide a service for students to communicate with one another, whether it be directly to another student or in a group of students. The app also allows users to create events for any group chats they are a part of, with a calendar connected to each group which will display these events. The application will only be available on Android devices when first released but there is a plan for it to be released on other systems post release. The applications is being built using Android Studios and will be coded in Java, XML and NodeJS.

1.1 Background

I came up with the idea for my project after noticing that there are not many ways for people in a class to be able to communicate with each other outside the college. Not everybody uses Facebook and the college don't have a service for students to talk to each other. This can be a problem when working in a group project and one or two members cannot be contacted because they do not have a Facebook or Twitter account, and no one else in the group has their contact details. Another problem I've noticed is that if the class wants to organise something or are making a decision it would normally be done on Facebook, these means members of the class who are not on Facebook miss out on some decision-making for something which affects them. I want to build an app that all students use in college to get rid of those problems.

1.2 Aims

The objective of my project is to build a social application for students to be able share information with each other, have a discussion about classes, organise and create events for a group, or just talk with their fellow students.

The app will allow users to communicate with each other either through group chats or by direct messaging each other. Users can create group chats and add specific users to the chat They will have the ability to search through users to find who they want to add Each chat room will have its own interactive calendar where events, such as study sessions, can be created and displayed on.

1.3 Technologies

The application was developed using Android Studio as the development environment. It will be coded using java, with XML used to design the user interface on the app and NodeJS is being used to handle the chat backend.

The application utilises a SQLite database which is hosted on an external web server and has PHP scripts which handle requests.

Pusher is used to handle real-time events on the application.

1.4 Structure

The *System* chapter explains the functional and non-functional requirements for the application. It also covers the design of the app and how each feature was implemented. There are also images of the GUI with explanations of what is happening on the screen. How the app was tested is also covered in this chapter.

The *Conclusion* will explain how I feel after completing the project, what I learned from this experience, and the problems I faced throughout the development process. It also covers the advantages and disadvantages of the app.

The *Further Development* chapter will describe the future plans I have for the application.

2 System

2.1 Requirements

2.1.1 Functional requirements

Registration

Description & Priority

The user shall have the ability to register an account. This is a high priority as users will be required to register an account in order to be able to use the service.

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

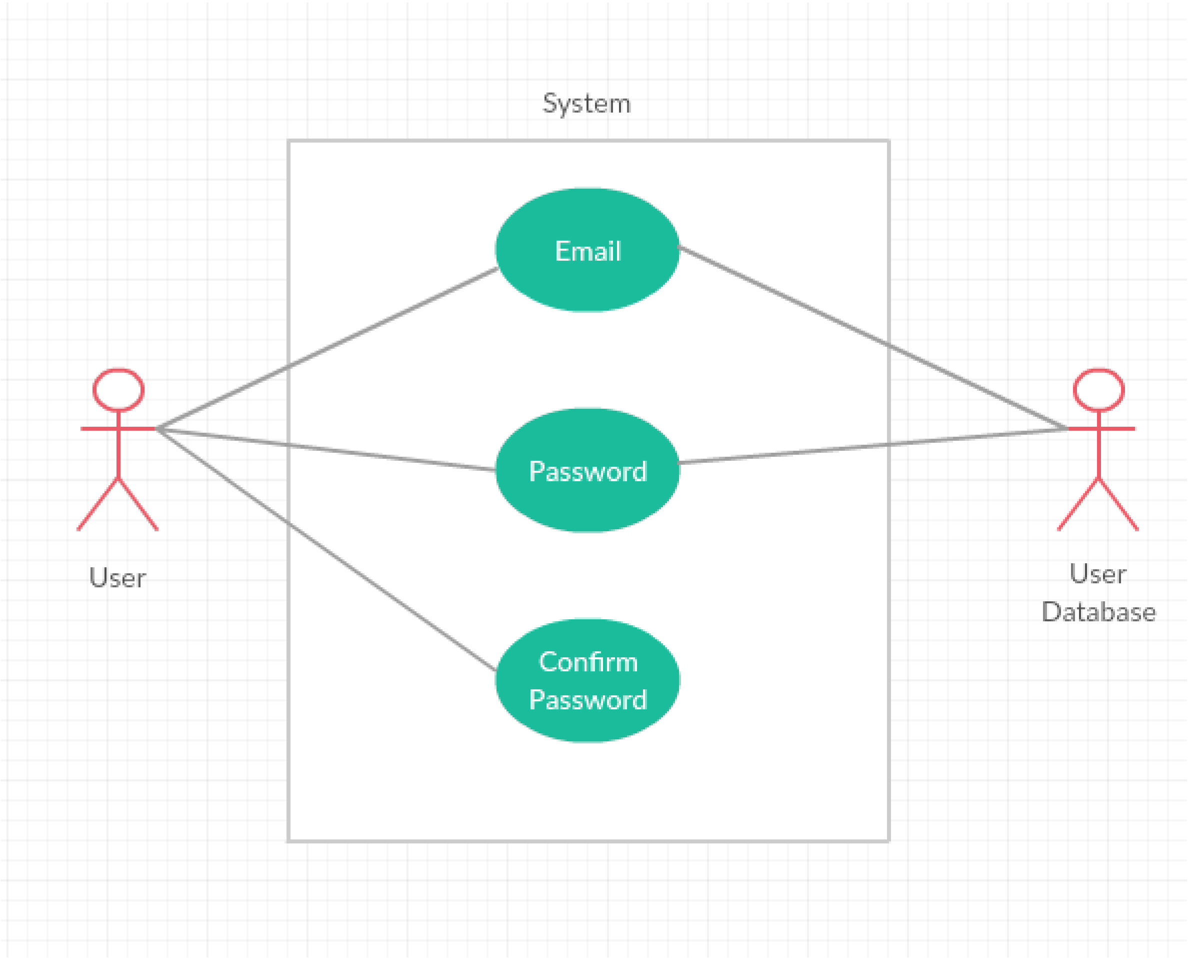
Scope

The scope of this use case is to allow a user to register an account on the application using an email and password which will be stored on a database

Description

This u0se case describes how a user can register an account on the application and how the account information is stored on a database

Use Case Diagram



Flow Description

Precondition

The application is opened with no user logged in.

Activation

This use case starts when the <user> clicks on the registration button on the homepage

Main flow

1. The system identifies the user has clicked the registration button and opens the registration page

2. The <user> clicks on the email textfield and enters an email address
3. The system verifies the email address is valid(See E1)
4. The <user> clicks on the password field and enters a password
5. The system verifies the password is of suitable length(See E1)
6. The <user> clicks the register button(See A1)
7. The <database> generates a UserID for the <user> and stores all the user details into the correct database table
8. The <user> is logged in

Alternate flow

A1: User already exists

1. The <user> clicks the register button
2. The system recognises the user already exists on the database
3. The system verifies the details are correct(See E1)
4. The use case continues at position 8 of the main flow

Exceptional flow

E1: Invalid Details

1. The <user> enters invalid details into the email or password field
2. The system recognises the invalid details and displays an error message
3. The use case continues at position 2 of the main flow

Termination

The system presents the homepage for a logged in user

Post condition

The system goes remains active

Log in

Description & Priority

The user shall have the ability to log in with a registered account. This is a high priority as users will be required to be logged in in order to be able to use the service.

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

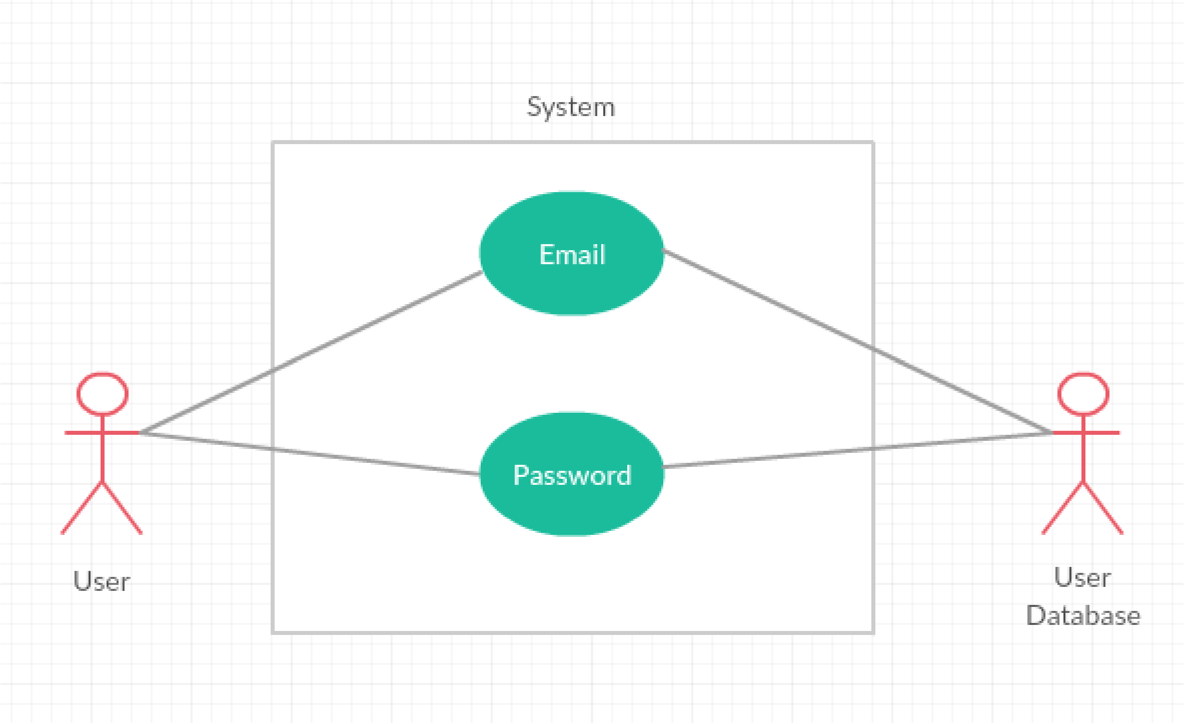
Scope

The scope of this use case is to allow the user to log in to the application using a registered account

Description

This use case describes how a user can log in to the application and how the system handles a login

Use Case Diagram



Flow Description

Precondition

The application is opened with no user logged in. The <user> has previously registered an account

Activation

This use case starts when the <user> clicks the login button on the homepage

Main flow

1. The system identifies the user has clicked the log in button and opens the login page
2. The <user> clicks on the email textfield and enters the email address they registered with
3. The <user> clicks on the password field and enters a password
4. The system verifies the password is of suitable length(See E1)

5. The <user> clicks the login button
6. The system verifies the user is registered on the database(See E2)
7. The <user> is logged in

Exceptional flow

E1: Invalid

1. The <user> enters invalid details into the password field
2. The system recognises the invalid details and displays an error message
3. The use case continues at position 2 of the main flow

E2: Account not registered

1. The <user> enters an email address that is not registered
2. The <user> enters in a password and clicks login
3. The system returns an error message
4. The use case continues at position 2 of the main flow

Termination

The system presents the homepage for a logged in user

Post condition

The system remains active

Message

Description & Priority

The user will have the ability to send messages to other users, whether it be through direct message or group chat. This is a high priority requirement as it is the main function of the application

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

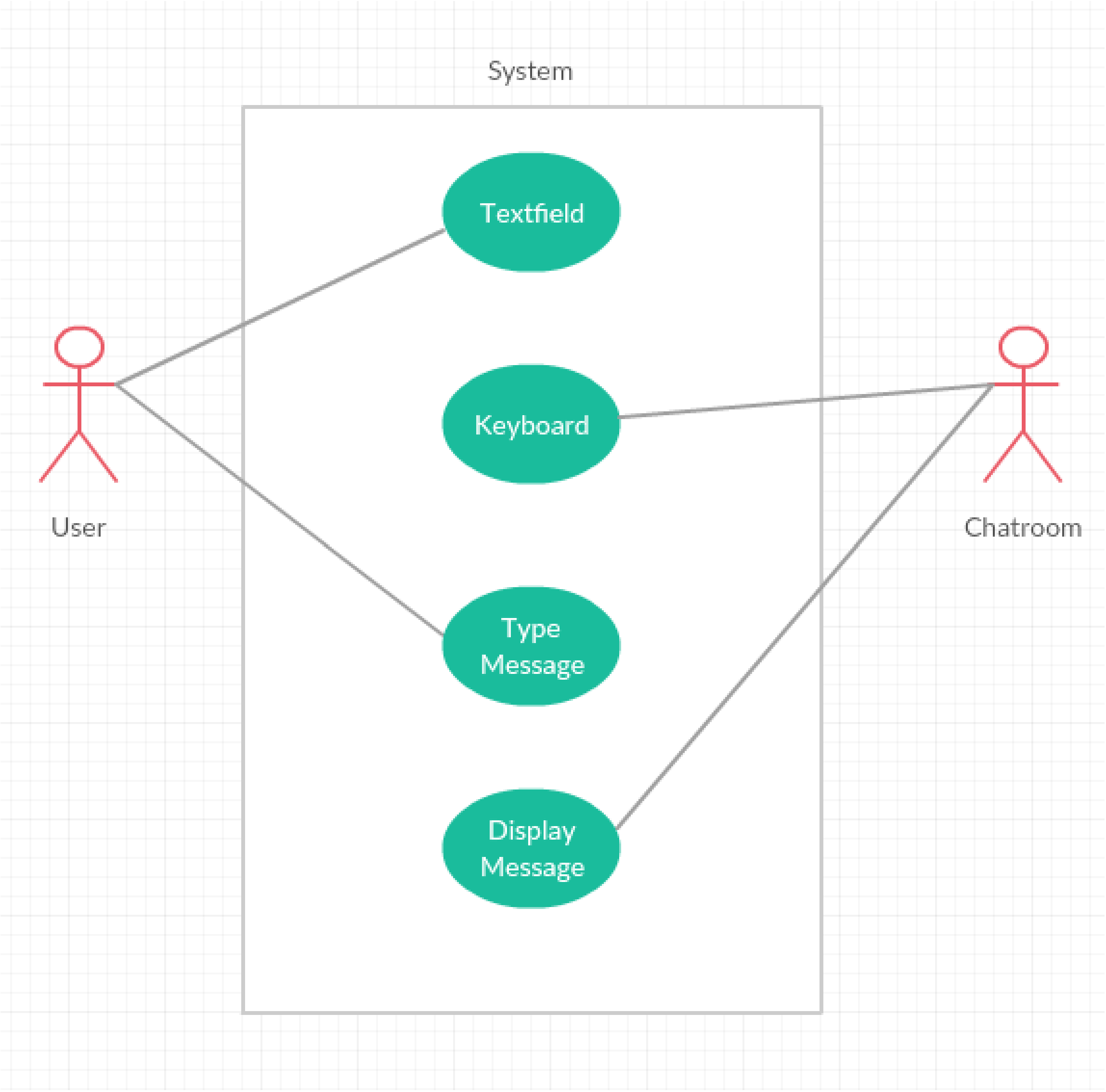
Scope

The scope of this use case is to allow a user to send a message using the application

Description

This use case describes how the user sends a message to another user and how the system handles sending a message

Use Case Diagram



Flow Description

Precondition

The system is active and a user is logged in

Activation

This use case starts when the <user> has opened a chat with another user or a group chat

Main flow

1. The <user> clicks the textfield on the bottom of the screen
2. The system opens up the native keyboard
3. The <user> types a message and clicks send
4. The system processes the message and sends it to the other user(s) (See E1)
5. The message appears for the other user(s)

Exceptional flow

E1: No Internet Connection

1. The system tries to process the message
2. No internet connection can be found
3. The <user> receives an error message prompting them to retry
4. The use case continues at position 1 of the main flow

Termination

The system presents the message on screen

Post condition

The system remains active

Create Chat

Description & Priority

The user will have the ability to create group chats on the application. This is a medium priority task as it is not essential to the application

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

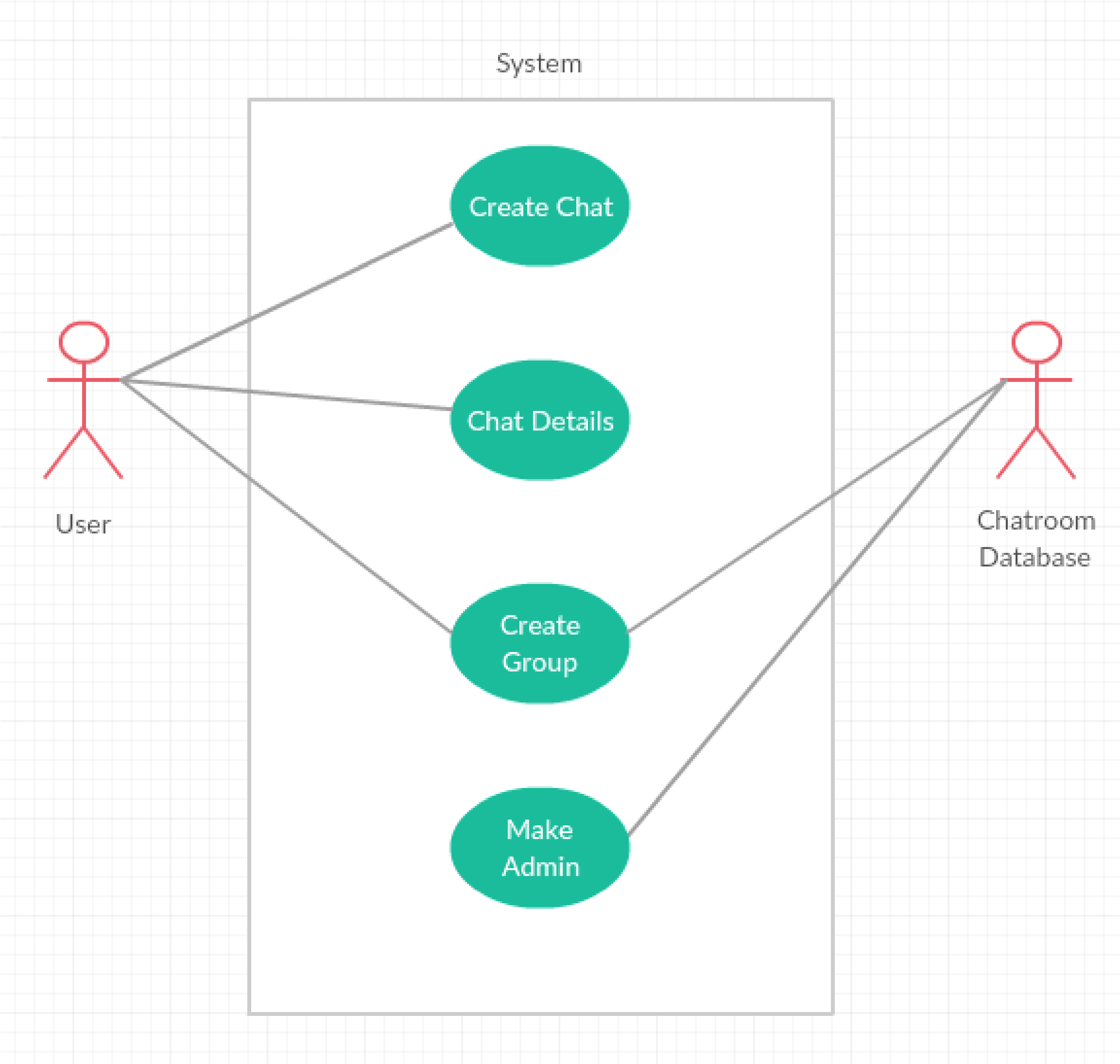
Scope

The scope of this use case is to allow a user to create a group chat and give that user admin rights to the chat

Description

This use case describes how the user can create a group chat and how the system gives that user admin rights for that chat

Use Case Diagram



Flow Description

Precondition

The system is active with a user logged in

Activation

This use case starts when the <user> clicks the create chat button in the menu

Main flow

1. System displays window to create chat
2. <User> enters in chat details and clicks create button
3. System creates group and adds it to database
4. System makes changes to database to make user admin of chat

Termination

The system presents a new chatroom

Post condition

The system goes remains active

Create Event

Description & Priority

The user will have the ability to create events for a chat group. This is a medium requirement for the application as it is not essential but is a release feature

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

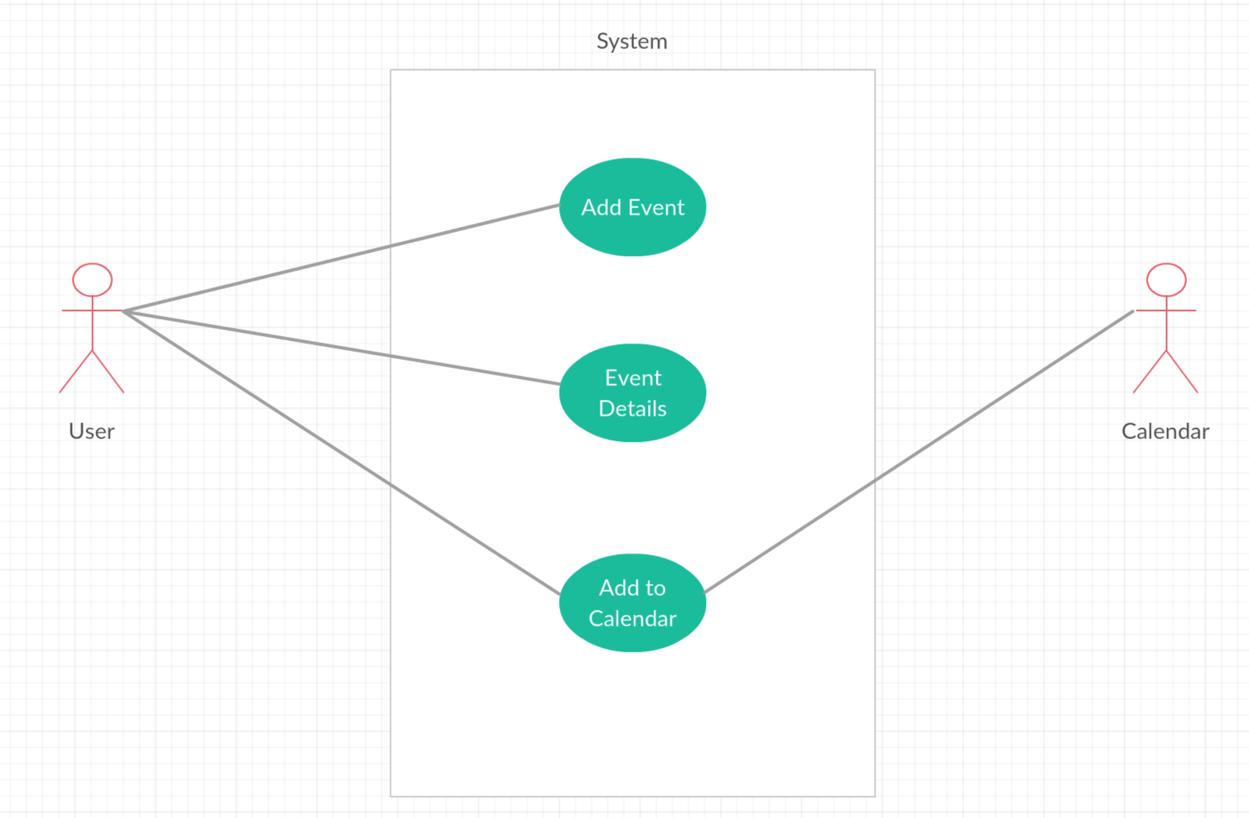
Scope

The scope of this use case is to allow users to create events for chat groups using the interactive calendar attached to each group

Description

This use case describes how a user can create a group event

Use Case Diagram



Flow Description

Precondition

The system is active and the user is logged in

Activation

This use case starts when a <user> clicks create event in a chat room

Main flow

1. The system displays the page for creating an event

2. The <user> enters the details for the event and selects a date.
3. The <user> clicks the create button
4. The system adds the event to the calendar

Termination

The system presents calendar with the event listed on it

Post condition

The system remains active

Add to chat

Description & Priority

Users will have the ability to add other users to a group chat. This is a medium priority task as group chats are only medium priority.

Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

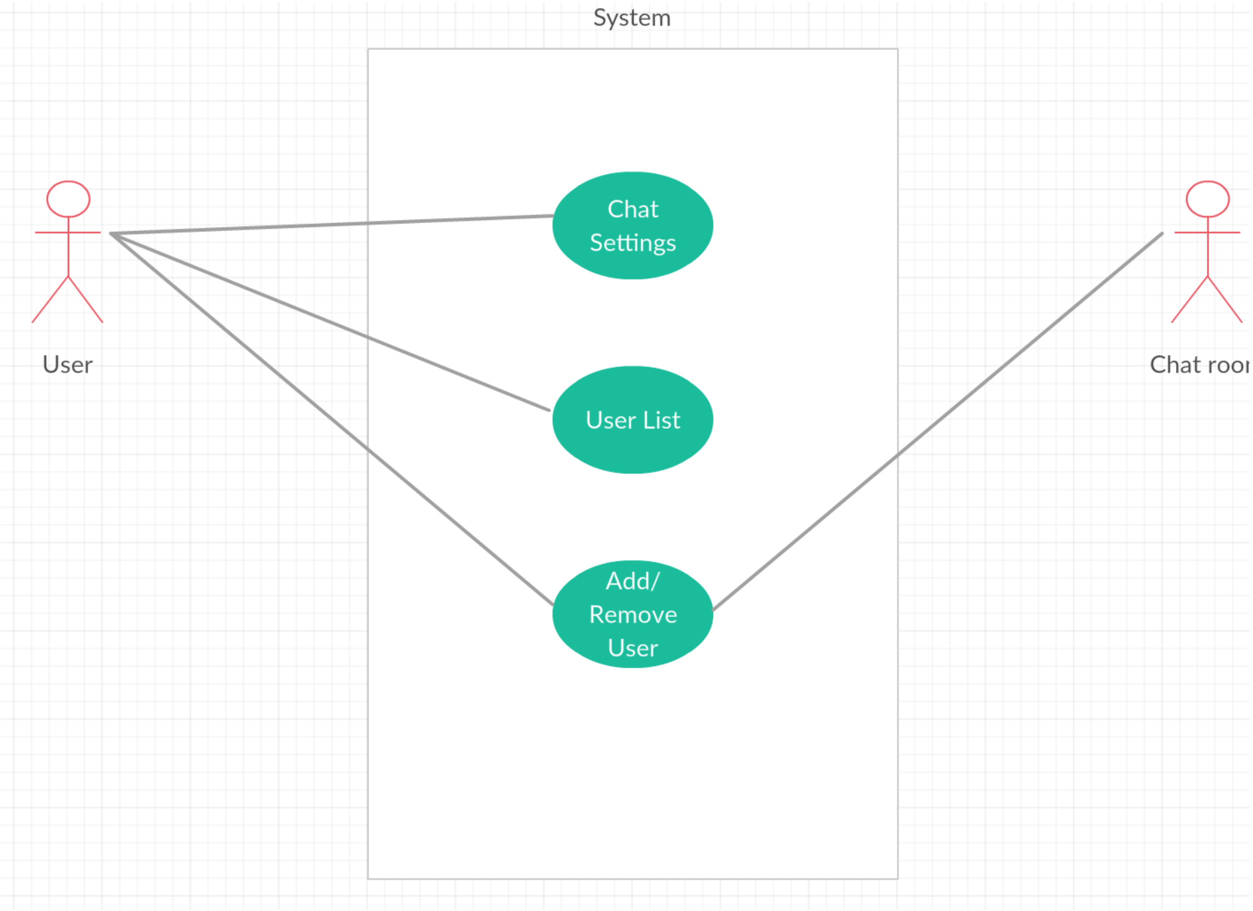
Scope

The scope of this use case is to allow users to add other users to a group chat

Description

This use case describes how a user can add or remove another user from a chat

Use Case Diagram



Flow Description-

Precondition

The system is active

Activation

This use case starts when an <User> clicks the settings button on a group chat

Main flow

1. The <User> selects add settings menu
2. The system opens a list of users

3. The <user> searches through other users checking the checkbox for the users they selected
4. The <user> clicks the add/remove button
5. The system performs the correct task based on whether the user selected add or remove
6. The system displays a message on the chat wall informing the other users that a user has been add/removed

Termination

The system presents the chat room with the new users added or the user has been removed

Post condition

The system remains active

2.1.2 Data requirements

The app will use a SQLite database which will store user information which will be used for the log in feature and displaying other user's names on the app.

2.1.3 User requirements

Availability – The user will require an internet connection in order for them to be able to use the services provided by the app. The application will only be available on mobile devices.

Security – A user will require an account in order to be able to use the application. This account will be stored on a database.

Reusability – Users will be required to return to the app to check on messages and upcoming events.

2.1.4 Usability requirements

Portability – The application will only be available on Android devices.

Reliability – The application should be able to withstand a large volume of users being online. It should not crash and performance should not be massively affected by the large number of users.

Performance/Response time – The system should have a quick response time from when a message is sent to when it is received. The application should also be fast when the user is navigating through different pages or trying to login or register.

2.1.5 Environmental Requirements

Android Device – The application is required to be run on an Android device in order for it to work.

Internet Access – The app is required to have access to the internet in order for the majority of its features to work.

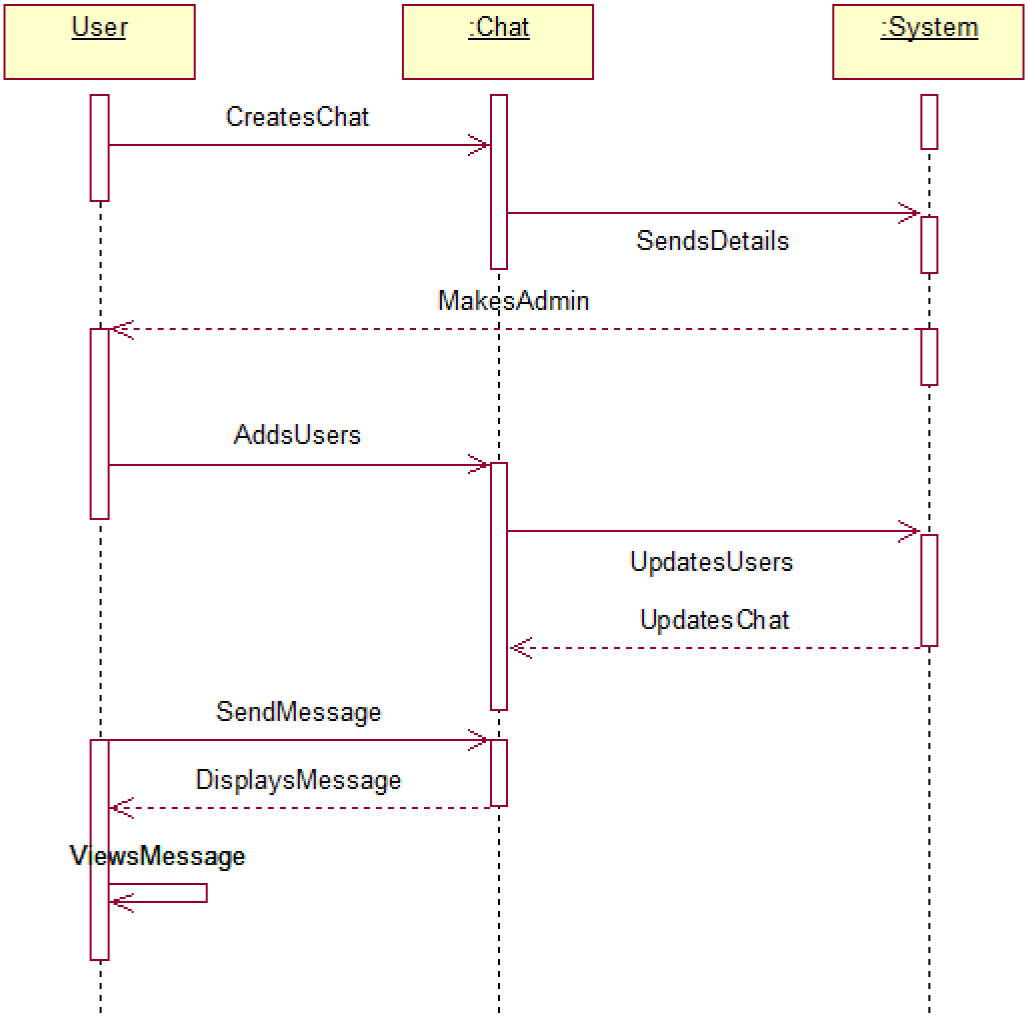
Windows – The application was developed using Android Studio on a Windows device

2.2 Design and Architecture

The application is being built using Java and XML, as well as PHP to process requests from the SQLite database. I chose these languages because I have used them before and that made me feel more confident in my ability to code the features in the application. The messaging feature was built using Node.js to power the chat back-end, with java used to program the rest.

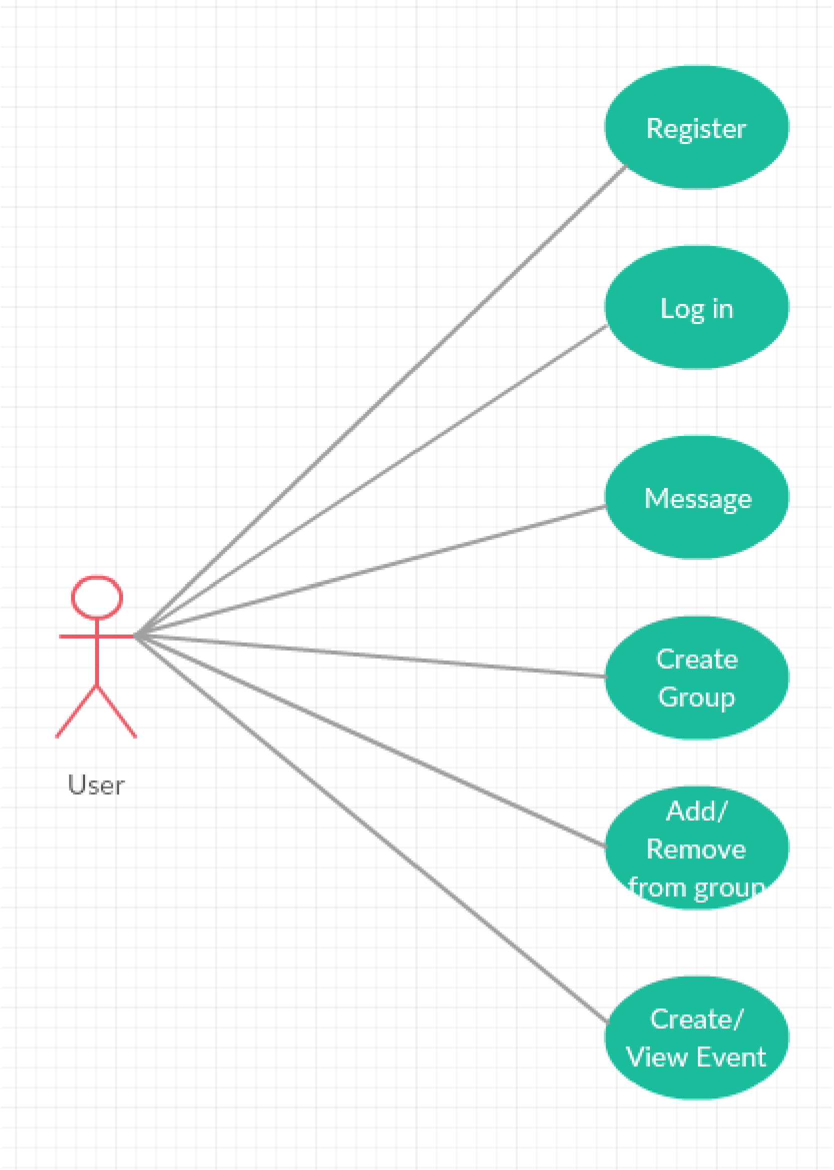
Sequence Diagram

Progression through creating a chat to sending a message.



Use Case Diagram

This diagram displays how a user can interact with the features in the application.



2.3 *Implementation*

The purpose of this section is to describe the technologies used in the implementation of the Android application. It will also cover the methodologies used in the implementation.

2.3.1 Technology Overview

The approach I took when building the android application was the native app development approach. This means the app was created using Google's preferred coding language, java, and their own building environment, Android Studio.

2.3.2 Technologies

2.3.2.1 Android Studio

Android Studio is the official IDE for Android app development. It is powered by IntelliJ IDEA, which offers powerful developer tools and a powerful code editor. As well as that, it also has a Gradle-based build system, and code templates to help build common app features. I used Android Studio 1.5.1 to develop the application.

2.3.2.2 Java

Java is a general-purpose computer programming language. It is concurrent, class based, object oriented, and designed to have as few implementation dependencies as possible. It is the default language used in Android development, since each activity is created with a java class.

2.3.2.3 XML

Extinsible Markup Language (XML) defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is used in Android development to design the layout of an application.

2.3.2.4 SQLite

All data for my application is stored on a SQLite database that is held on an external website (campustalk.gear.host).

2.3.2.5 PHP

PHP is a server-side scripting language designed for web development. I used it to connect my application to the SQLite database and to run functions on the database.

2.3.2.6 NodeJS

NodeJS is an open-source, cross-platform runtime environment for developing server-side web applications. For my application, NodeJS was used to handle the chat backend using the Express framework built for NodeJS.

2.3.2.7 Pusher

Pusher is a simple hosted API for quickly, easily and securely integrating real-time bi-directional functionality via WebSockets to web and mobile apps, or any other Internet connected device. This is used in the application to handle the real-time events which occur when messaging.

2.3.3 Procedures

2.3.3.1 User Registration and Login

For the registration and login features of the application a PHP script (DBFunctions.php) was written to handle functions such as storing a user on the database, retrieving user info, and checking if certain users already exist on the database. This script was then used by another script (index.php) to either register a user or log them in. Here are a few code snippets that show these functions:

Existing User

```
public function isUserExisted($email){
                    $result = mysql_query("SELECT email FROM users WHERE
email = '$email'");
                    $no_of_rows = mysql_num_rows($result);
                    if($no_of_rows > 0){
                            return true;
                    }else {
                            return false;
```

```
            }
        }
```

Store User

```
   public function storeUser($name, $email, $password){
            $uuid = uniqid("",true);
            $hash = $this->hashSSHA($password);
            $encrypted_password = $hash["encrypted"];
            $salt = $hash["salt"];
            $result = mysql_query("INSERt INTO users(unique_id, name, email, encrypted_password, salt, created_at) VALUES('$uuid','$name', '$email', $encrypted_password', '$salt', NOW())");
            if($result){
                $uid = mysql_insert_id();
                $result = mysql_query("SELECT * FROM users WHERE userID = '$uid'");
                return mysql_fetch_array($result);
            }else{
                return false;
            }
        }
```

Retrieve User Details

```
public function getUserByEmailAndPassword($email, $password){
    $result = mysql_query("SELECT * FROM users WHERE email = '$email'")or die(mysql_error());
    $no_of_rows = mysql_num_rows($result);
    if($no_of_rows > 0){
        $result = mysql_fetch_array($result);
        $salt = $result['salt'];
        $encrypted_password= $result['encrypted_password'];
        $hash = $this->checkhashSSHA($salt, $password);
        if($encrypted_password == $hash){
            return $result;
        }else{
            return false;
        }
    }
}
```

The application has a java class for both features as well, RegistrationActivity.java and LoginActivity.java. Both classes use the Android Volley library to handle networking calls.

2.3.3.2 Messaging

The messaging feature required three java classes: ChatActivity.java, Message.java, and MessageAdapter.java.

The Message.java class is used to comprise the list of who sent the message, the message text and when the message was sent.

The MessageAdapter.java class is used by ChatActivity.java to change the list in Message.java into a set of views.

ChatActivity.java is used to display the messages on a single page and includes the code behind the send button and the text field.

2.4 Testing

Software testing is vital to any software development project. Therefore I spent a considerable amount of time testing my application.

2.4.1 Unit Testing

Unit testing was carried out on the completion of each function. This involved running my application on an Android device and monitoring the changes from the new code. I tested these on a variety of devices and on the emulator in Android Studio.

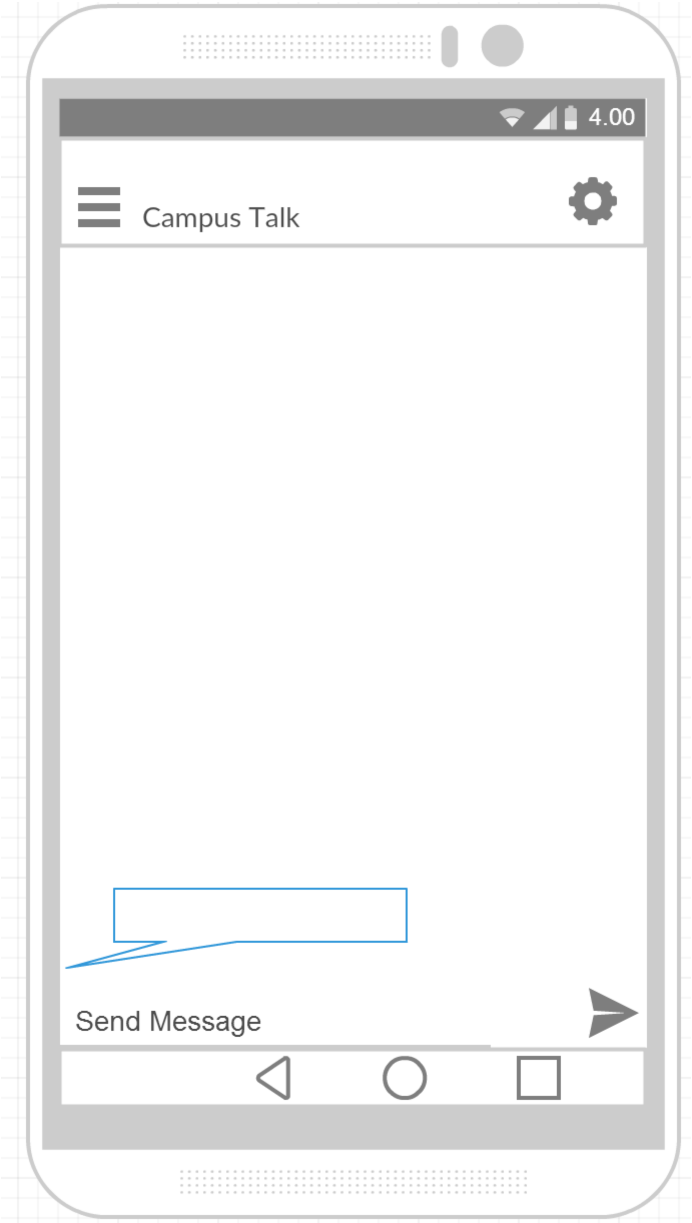
2.4.2 System Testing

For system testing I created a number of test cases which describe how each feature works and describe the steps taken to test the feature. These test cases can be found in the appendix of this report. I then got a number of people from the college to test the application for me, using said test cases as a guide for each feature. They would then notify me about any errors they found.

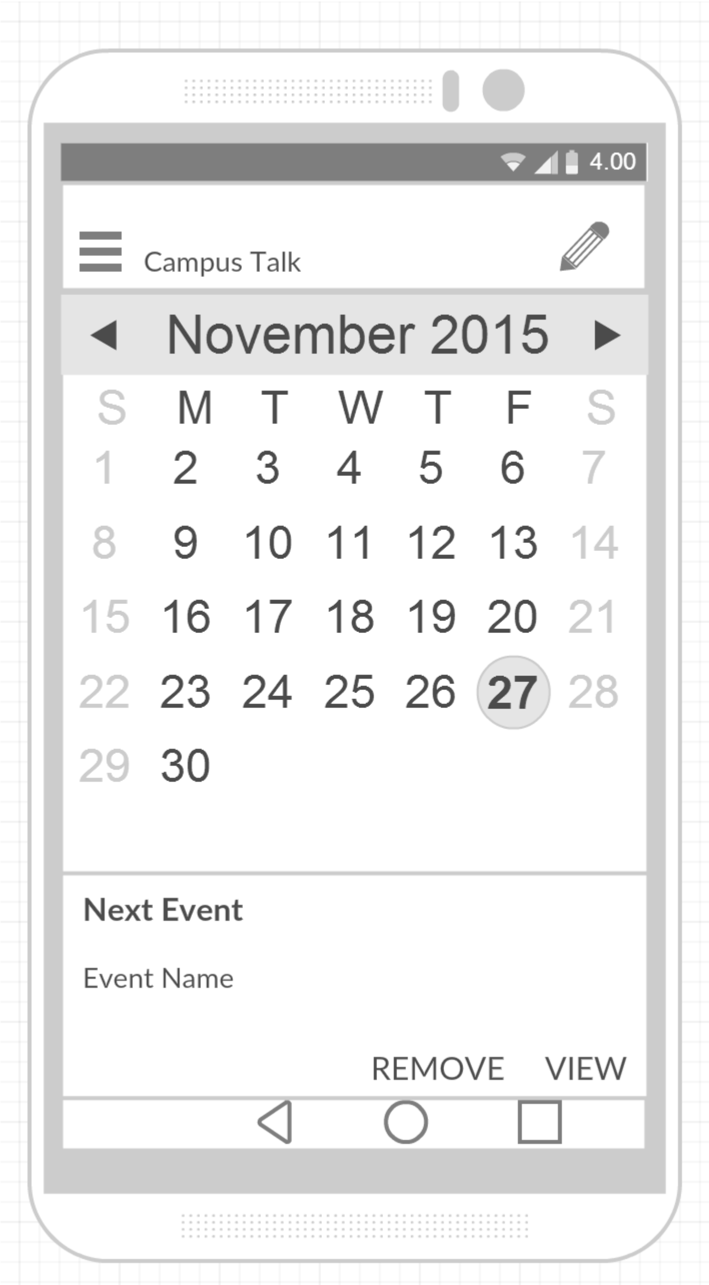
2.4.3 Backwards-Compatibility Testing

To test that the application was able to work on phones with different screens and older versions of Android I created multiple devices to use on the emulator in Android Studio. Each device was created with a different version of Android and a different screen size.

2.5 Graphical User Interface (GUI) Layout

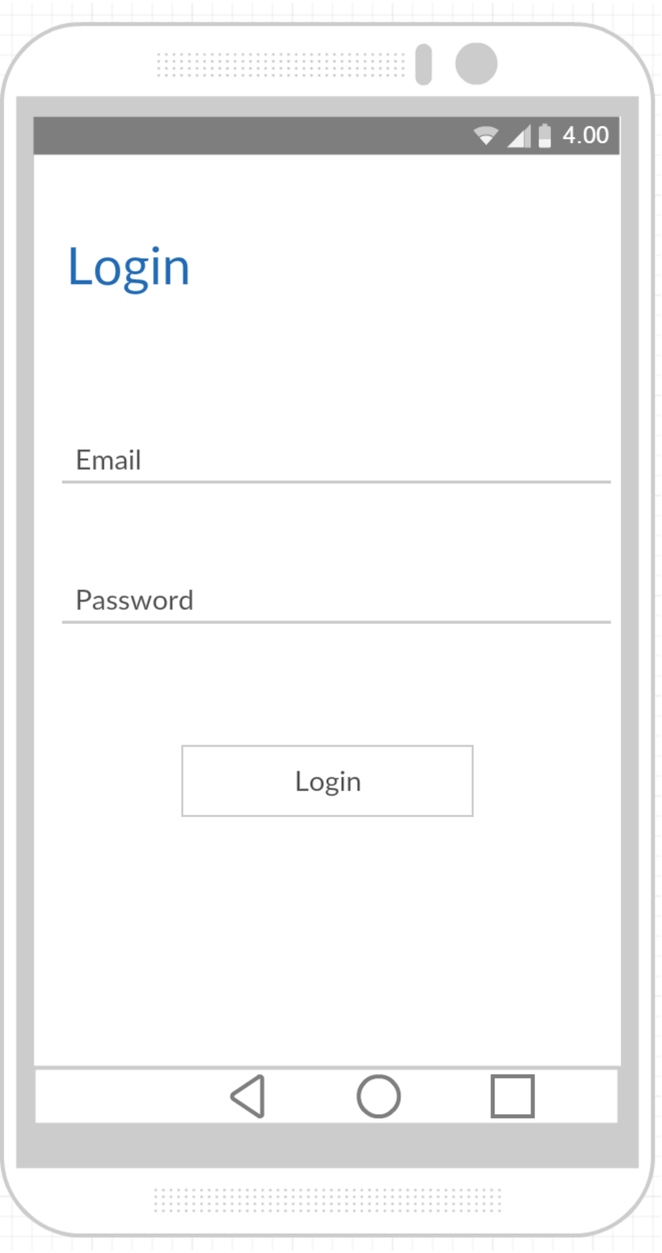


Chat Interface – This will be the interface for the chat rooms in the application. The user will click on the textfield at the bottom of the screen to start entering text. Once a user has sent a message it will appear in a bubble just above the textfield.

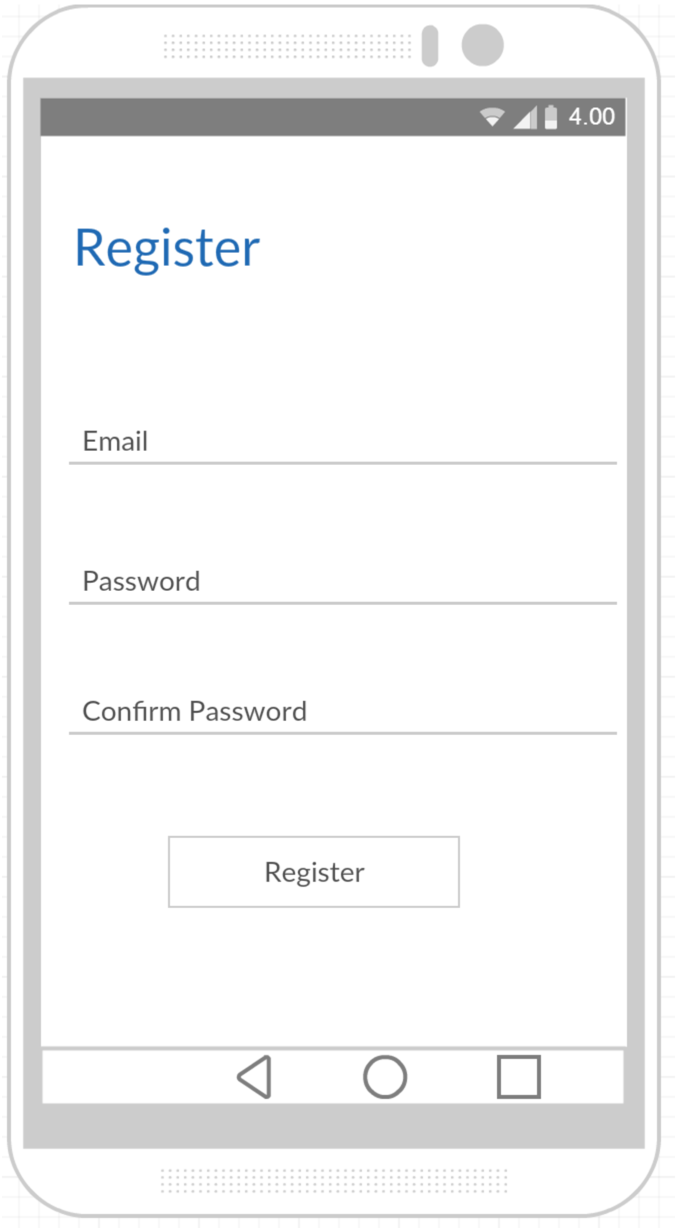


S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

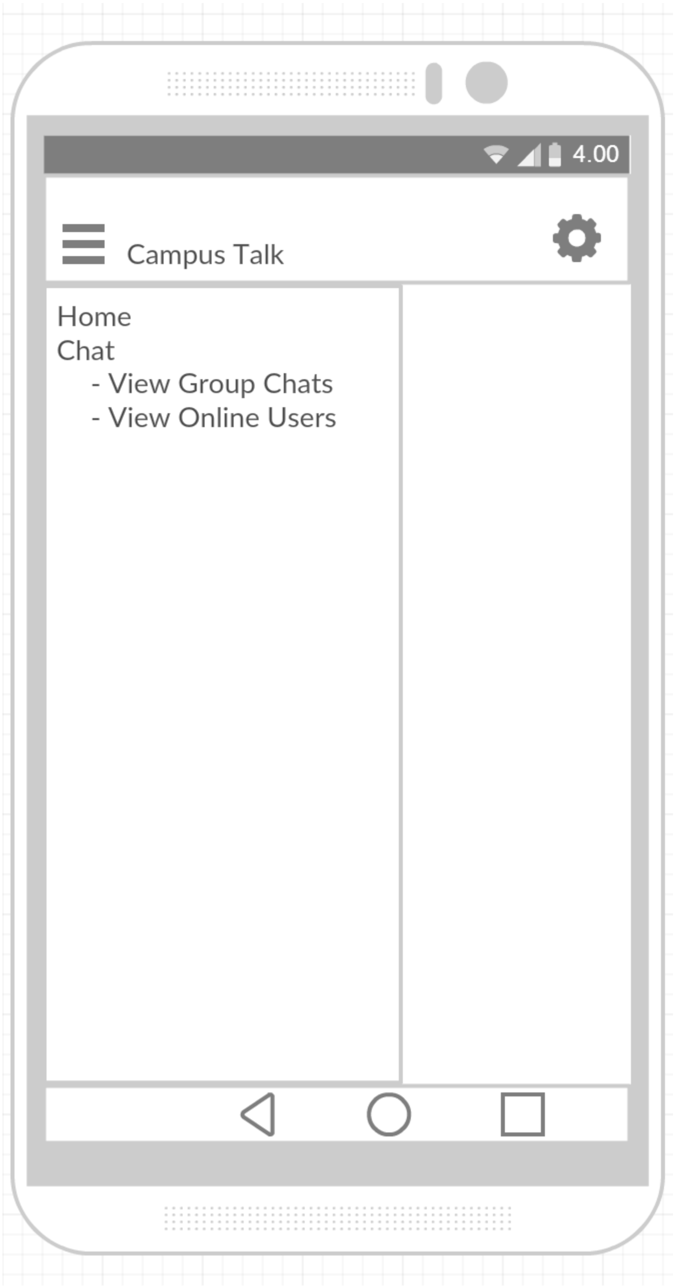
Calendar Interface – This will be the interface for the calendar connected to each chatroom. When opened it will display the current month with the current day highlighted. If there is an upcoming event for the chat group a notification will be displayed at the bottom of the screen with a few details of the upcoming event.



Login Interface – There will be a generic login form used in the application. The user will be required to enter an email and password in order to log in.



Registration Interface – There will be a generic registration form where the user is required to sign up with an email and password and must also confirm their password in order for the registration to be successful.



Side Menu Interface – The user activates the side menu by clicking a button in the top left corner of the screen. The side menu contains links to the home page and different chatrooms or users to chat with.

3 Conclusions

Overall I enjoyed my time developing this project. It has proved to be challenging and through my attempts to overcome those challenges, I have achieved a greater understanding of different programming languages and learnt new techniques in software development. Coming in to the project I had very little knowledge of PHP and NodeJS, but now I feel confident in my ability to work with these.

I enjoyed working on a messaging app because it was fun to messing with different layouts and designs I could have for the app. It is also one of the most popular areas to develop an app in with many similar apps such as Viber, WhatsApp, and Facebook Messenger available in the market.

My only regret is that I did not have enough time to implement all the features I wanted and that I was only able to develop on Android and could not work on other platforms like iOS.

Overall though, despite my disappointments, I am pleased with my experience developing my application.

4 Further development

In the future the app I intend on adding more features to the app, such as:

Emojis: Users will be able to use their favourite emojis when talking to each other on the app. This should help make the app seem friendlier and might encourage users to engage more in conversation on this app than other apps they use for messaging.

Profile Pictures: Users will be able to add a picture which will be displayed beside their name on the application.

Ability to send images/videos: Users will be able to send an image or a video as a message.

I also plan on building an iOS version of the application which will increase the customer base, and in turn the number of users using the application. This will lead to an increase in database requests which means the database will also have to be upgraded in the future.

5 Appendix

5.1 Project Proposal

Project Proposal

Campus Talk

Niall O’Connell, 12431302, x12431302@student.ncirl.ie

BSc (Hons) in Computing

Networking and Mobile Technologies

Date

28/09/15

Objectives

The objective of my project is to build a social application for students to be able share information with each other, have a discussion about classes, organise and create events for a group, or just talk with their fellow students.

The app will allow users to communicate with each other either through group chats (public or private) or by direct messaging to one another. Public group chats will be streamed by schools (computing, business), then by year and class with a general page where all students can chat to each other. Users will also be able to create private chat rooms where they can invite specific people to be a part of it. They will have the ability to search through users to find who they want to add and the creator of the group will also have the ability to remove people from the chat.

Each chat room will have its own interactive calendar where events, such as study sessions, can be created and displayed on. Calendars on the public chat rooms will be limited to events organised by the college or student's union.

Background

I came up with the idea for my project after noticing that there is no one way for everybody in a class to be able to communicate with each other outside the college. Not everybody uses Facebook and the college don't have a service for students to talk to each other. This can be a problem when working in a group project and one or two members cannot be contacted because they do not have a Facebook or Twitter account, and no one else in the group has their contact details. Another problem I've noticed is that if the class wants to organise something or are making a decision it would normally be done on Facebook, these means members of the class who are not on Facebook miss out on some decision-making for something which affects them. I want to build an app that all students use in college to get rid of those problems.

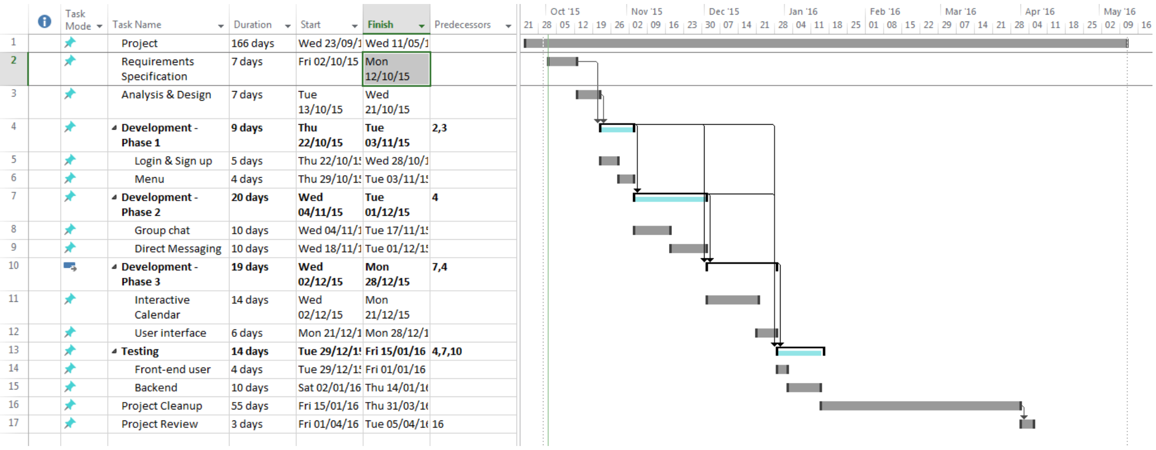
Technical Approach

My approach to the project will be to complete all documentation first to not be distracted and pulled away from work on the app when a document is due to be uploaded. Once I complete all the documentation I will then begin researching similar applications to see what I can do differently and to find any resources which may help me with my project. I will then begin to code the application building basic pages, such as login and sign up, first before moving on to the more difficult features. Once all the features are complete I will add the UI and send the app to be tested. After testing is complete I will go back and fix any bugs which were found during testing and test that feature again.

Special resources required

- Android smartphone
- 000webhost.com
- Android Studios

Project Plan



	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		📌	Project	166 days	Wed 23/09/1	Wed 11/05/1	
2		📌	Requirements Specification	7 days	Fri 02/10/15	Mon 12/10/15	
3		📌	Analysis & Design	7 days	Tue 13/10/15	Wed 21/10/15	
4		📌	**Development - Phase 1**	**9 days**	**Thu 22/10/15**	**Tue 03/11/15**	**2,3**
5		📌	Login & Sign up	5 days	Thu 22/10/1	Wed 28/10/1	
6		📌	Menu	4 days	Thu 29/10/1	Tue 03/11/1	
7		📌	**Development - Phase 2**	**20 days**	**Wed 04/11/15**	**Tue 01/12/15**	**4**
8		📌	Group chat	10 days	Wed 04/11/1	Tue 17/11/1	
9		📌	Direct Messaging	10 days	Wed 18/11/1	Tue 01/12/1	
10		➡	**Development - Phase 3**	**19 days**	**Wed 02/12/15**	**Mon 28/12/15**	**7,4**
11		📌	Interactive Calendar	14 days	Wed 02/12/15	Mon 21/12/15	
12		📌	User interface	6 days	Mon 21/12/1	Mon 28/12/1	
13		📌	**Testing**	**14 days**	**Tue 29/12/1**	**Fri 15/01/16**	**4,7,10**
14		📌	Front-end user	4 days	Tue 29/12/1	Fri 01/01/16	
15		📌	Backend	10 days	Sat 02/01/16	Thu 14/01/1	
16		📌	Project Cleanup	55 days	Fri 15/01/16	Thu 31/03/1	
17		📌	Project Review	3 days	Fri 01/04/16	Tue 05/04/1	16

Technical Details

- Java
- XML
- AndroidKickstartr

Evaluation

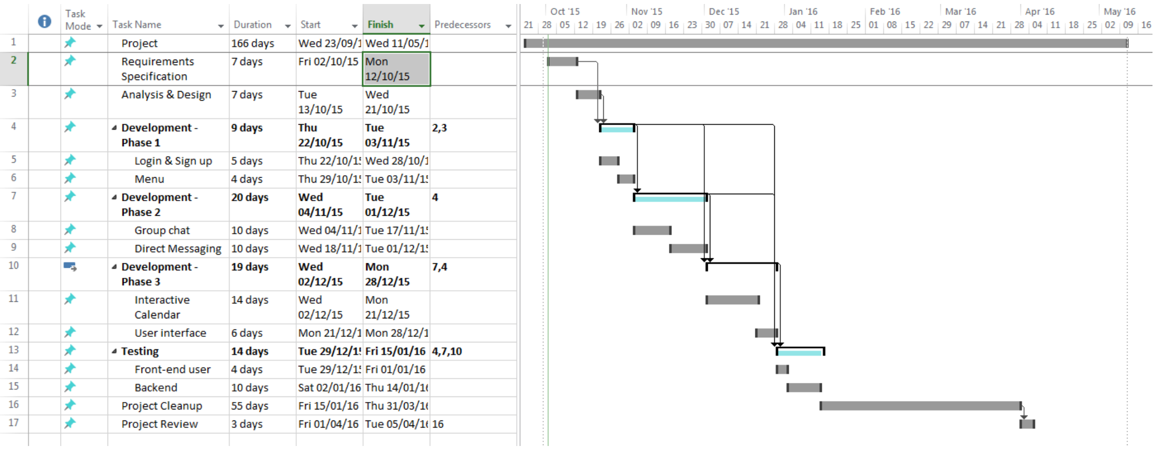
To test the end user experience of the application I will write a detailed description of each function, stating how it should work and look, and give the app to people with the descriptions of each function for them to test. If a problem is found they will report back with where the problem is, how serious it is and give me steps on how to reproduce the problem.

_Niall O Connell__02/10/15_____

Signature of student and date

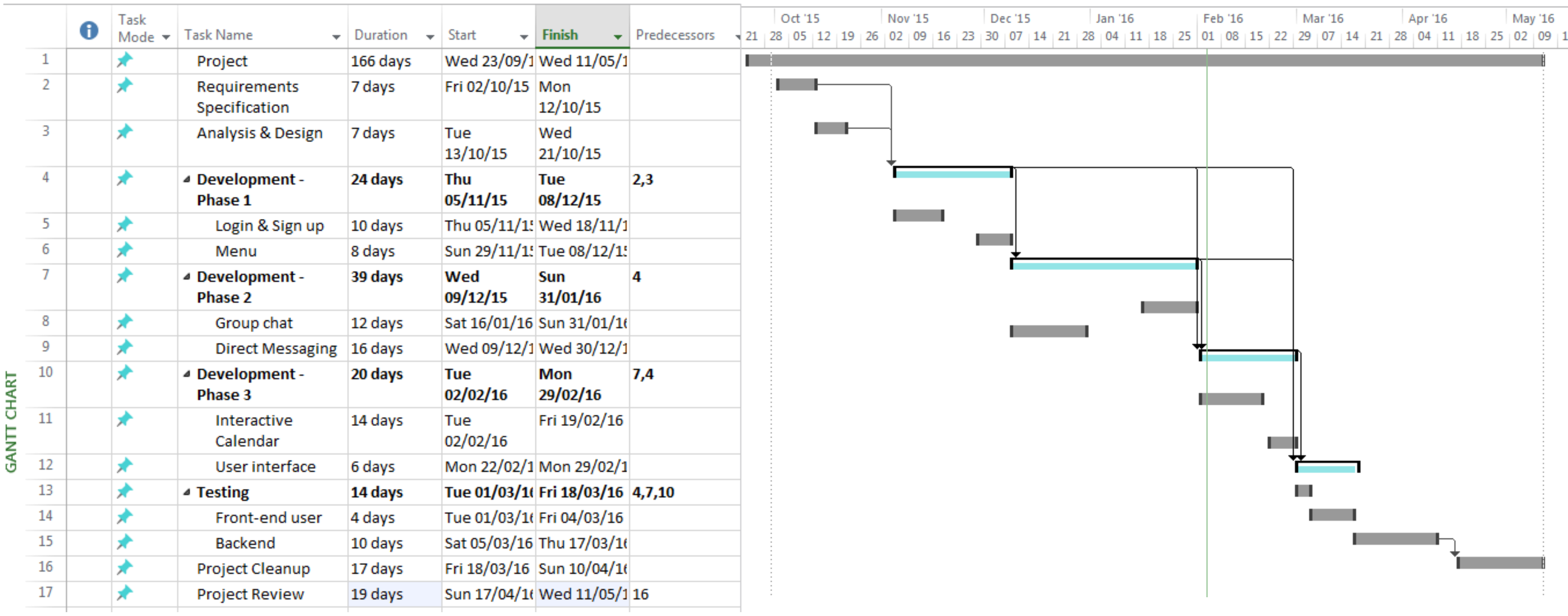
5.2 Project Plan

Old Plan



	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		📌	Project	166 days	Wed 23/09/1	Wed 11/05/1	
2		📌	Requirements Specification	7 days	Fri 02/10/15	Mon 12/10/15	
3		📌	Analysis & Design	7 days	Tue 13/10/15	Wed 21/10/15	
4		📌	**Development - Phase 1**	**9 days**	**Thu 22/10/15**	**Tue 03/11/15**	**2,3**
5		📌	Login & Sign up	5 days	Thu 22/10/1!	Wed 28/10/1	
6		📌	Menu	4 days	Thu 29/10/1!	Tue 03/11/1!	
7		📌	**Development - Phase 2**	**20 days**	**Wed 04/11/15**	**Tue 01/12/15**	**4**
8		📌	Group chat	10 days	Wed 04/11/1	Tue 17/11/1!	
9		📌	Direct Messaging	10 days	Wed 18/11/1	Tue 01/12/1!	
10		➔	**Development - Phase 3**	**19 days**	**Wed 02/12/15**	**Mon 28/12/15**	**7,4**
11		📌	Interactive Calendar	14 days	Wed 02/12/15	Mon 21/12/15	
12		📌	User interface	6 days	Mon 21/12/1	Mon 28/12/1	
13		📌	**Testing**	**14 days**	**Tue 29/12/1!**	**Fri 15/01/16**	**4,7,10**
14		📌	Front-end user	4 days	Tue 29/12/1!	Fri 01/01/16	
15		📌	Backend	10 days	Sat 02/01/16	Thu 14/01/1(	
16		📌	Project Cleanup	55 days	Fri 15/01/16	Thu 31/03/1(	
17		📌	Project Review	3 days	Fri 01/04/16	Tue 05/04/1(	16

New Plan



	ⓘ	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		📌	Project	166 days	Wed 23/09/1	Wed 11/05/1	
2		📌	Requirements Specification	7 days	Fri 02/10/15	Mon 12/10/15	
3		📌	Analysis & Design	7 days	Tue 13/10/15	Wed 21/10/15	
4		📌	**Development - Phase 1**	**24 days**	**Thu 05/11/15**	**Tue 08/12/15**	**2,3**
5		📌	Login & Sign up	10 days	Thu 05/11/1	Wed 18/11/1	
6		📌	Menu	8 days	Sun 29/11/1	Tue 08/12/1	
7		📌	**Development - Phase 2**	**39 days**	**Wed 09/12/15**	**Sun 31/01/16**	**4**
8		📌	Group chat	12 days	Sat 16/01/16	Sun 31/01/1	
9		📌	Direct Messaging	16 days	Wed 09/12/1	Wed 30/12/1	
10		📌	**Development - Phase 3**	**20 days**	**Tue 02/02/16**	**Mon 29/02/16**	**7,4**
11		📌	Interactive Calendar	14 days	Tue 02/02/16	Fri 19/02/16	
12		📌	User interface	6 days	Mon 22/02/1	Mon 29/02/1	
13		📌	**Testing**	**14 days**	**Tue 01/03/1**	**Fri 18/03/16**	**4,7,10**
14		📌	Front-end user	4 days	Tue 01/03/1	Fri 04/03/16	
15		📌	Backend	10 days	Sat 05/03/16	Thu 17/03/1	
16		📌	Project Cleanup	17 days	Fri 18/03/16	Sun 10/04/1	
17		📌	Project Review	19 days	Sun 17/04/1	Wed 11/05/1	16

5.3 *Monthly Journals*

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: September

My Achievements

This month I was able to decide on an idea for my project and then completed my project proposal, which was then uploaded to Moodle. I also began researching similar apps to get ideas on how I could build my app and what I could do to make it different. I looked at a few libraries which could be used for the app and looked at some tutorials for Android Studios.

My Reflection

I was happy with the idea for my app. I feel it's a good idea and it will be a good challenge for me in terms of my coding ability and hopefully I will enjoy building the application.

I feel I struggled a bit with the project proposal. I found it hard to explain my idea and had not thought about certain aspects of the project which I was being asked to explain. I feel I came up with my idea a bit late which left me with not a lot of time to think of these things.

Intended Changes

Next month I plan on completing the documentation for my project (requirement specification & analysis and design). This will leave me with a lot more time to put thought into what I am writing and if completed as planned I will have no distractions from coding the application.

I realised I need to get better at my time management for this project if I have any hope of completing everything as planned.

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

Reflective Journal

Student name: Niall O'Connell

Programme: BSc in Computing

Month: October

My Achievements

This month most of my time was spent working on other projects although I had spent some time working on the requirements specification for the project. I also had my first meeting with my project supervisor. In the meeting we spoke about my project idea and looked for ways I could improve on some of my ideas for the project.

My Reflection

I was not happy with my work this month as I feel I did not complete enough and am starting to fall behind on my original timescale. I was also unhappy with my time management as I spent too much time working on other projects I did not leave myself with a lot of time to work on the requirements specification and no time was spent coding.

Intended Changes

Next month I plan to start coding the application while also working on the analysis and design document. I will also finish the requirements specification and upload it to moodle.

Other projects are becoming more of a priority at the moment as deadlines are closing in but I hope I can still manage to split my time between them to leave me with time to work on my application.

Supervisor Meetings

Date of Meeting: 20/10/15

Items discussed: Project Idea

Action Items: Research ways of integrating programs such as Moodle into project

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: November

My Achievements

This month I tidied up my Requirements Specification document and worked on the Analysis and Design document which is due at the start of December. I also began work on the chat function of my application using the Firebase API.

I also researched different APIs and libraries that I could use for the chat function of my application.

My Reflection

I was happy with what I got done this month in terms of the amount of work done. I do feel that the Analysis and Design document could be cleaned up to look better. I’m still not too happy with the amount of coding I am getting done.

Intended Changes

Next month I plan to finish work on the chat feature and begin working on the login and registration for the application. I will also look into ways of creating the interactive calendar planned to be included in the application.

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: December

My Achievements

This month I worked on the chat, login and registration features of my application. I have managed to nearly finish them but have started spending more time studying for exams plus I am a lot busier in my job this month so a lot of my time is being taken up there.

My Reflection

I was happy with what I got done but was hoping to have more done before I started studying for the exams. I was also disappointed I did not finish my work on the chat feature for the application.

Intended Changes

Next month I plan to finish work on the chat, login, and registration and begin preparing for the mid-point presentation in February.

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: January

My Achievements

This month I finished my work on the register and login sections of the app, but unfortunately ran into a problem with the chat feature. I also worked on and finished the report for the mid-point presentations.

My Reflection

I was happy with finally finishing the register and login sections of the app, and that I was able to complete the mid-point report.

I was upset with the error with the chat as it may affect my ability to demonstrate the application for my mid-point presentation

Intended Changes

I intend to fix the error with the chat feature and begin working on the calendar feature of the application.

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: February

My Achievements

This month I did my mid-point presentation and I continued working on the chat feature of my application, having changed how it will be implemented. I am now using Pusher as my library and coding it with both NodeJS and java. It is mostly done with just a few more things needed in the server file. I also did the 30 – 40 word description of my project for the booklet for the project showcase in May.

My Reflection

I was happy with what I got done this month in terms of the amount of work done. I feel that the app is starting to take shape and will hopefully be able to test the chat feature next month. I was a bit upset with the result from my mid-point presentation as I felt I had done enough to earn a higher grade.

Intended Changes

Next month I plan to finish work on the chat feature and begin work on creating groups, events and the calendar for the app.

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

Reflective Journal

Student name: Niall O’Connell

Programme: BSc in Computing

Month: March

My Achievements

In March, I finished the events and calendar for the app and was halfway through creating groups before stopping work on the app to study for my exams. I also made my poster for the showcase in May and began working on parts of my technical report.

My Reflection

I was happy with what I got done this month in terms of the amount of work done. I feel that the app is starting to take shape. I was a bit disappointed I could not get more done before I stopped working to study for the exams.

Intended Changes

Next month I plan to finish work on the app, the technical report, and prepare for my presentation in May

Supervisor Meetings

Date of Meeting:

Items discussed:

Action Items:

5.4 Test Cases

	Registration – Success
Overview	Users should be able to register in the application using an email address, while also being required to enter a password and their name
Test Steps	1. Open the application 2. Click the sign up button 3. Enter a name 4. Enter an email address 5. Enter a password 6. Click Register button
On Completion	User will be brought to the login screen

	Registration – Fail
Overview	The app should be able to handle errors that may occur when a user is trying to register
Test Steps	1. Open the application 2. Click the sign up button 3. Enter a name 4. Enter an invalid password (e.g. test@@test.com) 5. Enter a password 6. Click Register button
On Completion	User will see a pop up message telling them the email address is invalid

	Login - Succes
Overview	Users should be able to log in to the app using the same details they registered with
Test Steps	1. Open the application 2. Click the login button 3. Enter a previously registered email address 4. Enter the password associated with the email address 5. Click Log In button
On Completion	User will be logged in to the application

	Login – Fail
Overview	The app should be able to handle any errors that may occur when a user is trying to log in
Test Steps	1. Open the application 2. Click the login button 3. Enter a previously registered email address 4. Enter an incorrect password 5. Click Log In button
On Completion	A pop up message will be displayed saying the email and password do not match

	Send Message
Overview	The main feature of the app is sending messages to other users
Test Steps	1. Log in to the app 2. Select a user to send a message to 3. Enter some text in the text field on the bottom of the screen 4. Click Send button
On Completion	The message will be displayed at the top of the screen with the user's name above it

	Create Event
Overview	Users should be able to create events in their group chats
Test Steps	1. Log in to the app 2. Open/create a group chat 3. Click the calendar button on the top right of the screen 4. Click the create event button 5. Enter event details 6. Click Create Event
On Completion	Event will be created and displayed on calendar

5.5 *References*

References

Developer.android.com. (2016). *Android Studio Overview | Android Developers*. [online] Available at: http://developer.android.com/tools/studio/index.html [Accessed 10 May 2016].

Expressjs.com. (2016). *Express - Node.js web application framework*. [online] Available at: http://expressjs.com/ [Accessed 10 May 2016].

Gearhost.com. (2016). *.NET and PHP Cloud Hosting | GearHost*. [online] Available at: https://www.gearhost.com/ [Accessed 10 May 2016].

Gradle. (2016). *Gradle l Modern Open-Source Enterprise Build Automation - Gradle*. [online] Available at: http://gradle.org/ [Accessed 10 May 2016].

Nodejs.org. (2016). *Node.js*. [online] Available at: https://nodejs.org/en/ [Accessed 10 May 2016].

Pusher.com. (2016). *Pusher | Leader In Realtime Technologies*. [online] Available at: https://pusher.com/ [Accessed 10 May 2016].